Quelle: http://mein-javablog.de/ bearbeitet von Jana Rau 08.2019

Speicherverwaltung in Java-Programmen

Java hat eine interne Speicherverwaltung. Eigentlich ist es nicht nur eine. Es sind mehrere.
Uns interessieren nur zwei.

Methoden und lokale Variablen werden auf dem Stack verwaltet.
Java-Objekte und Instanzvariable werden auf dem Heap verwaltet.

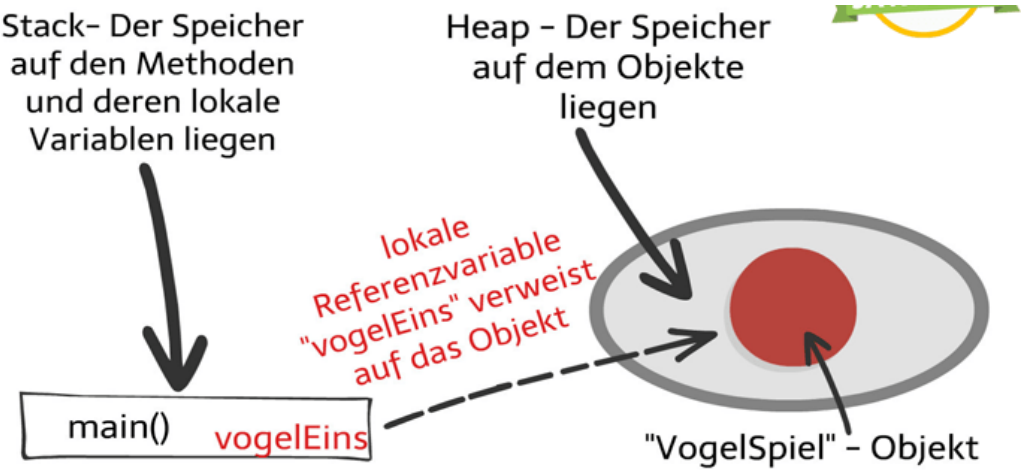
Wieso braucht Java zwei Speichersysteme?

Java-Objekte werden an einem anderen Ort gespeichert als Methoden oder lokale Variablen, weil ein Objekt meistens mehrere Variable umfasst. Es ist also **ein größerer Speicherbedarf** notwendig. Ein Objekt muss oft auch länger als eine Methode/lokale Variable im Programm gehalten werden.

Stelle dir vor, du hast ein Java-Spiel, bei dem du irgendwelche Vögel auf dem Bildschirm abschießen musst. Jeder Vogel, der getroffen wurde, verschwindet vom Bildschirm und dafür kommt ein neuer Vogel.

Wenn der Vogel vom Bildschirm verschwindet, verschwindet er dann auch aus dem Speicher? Oder kann es sein, dass sich der Speicher nach und nach mit Vogel-Objekten füllt, die niemand mehr braucht?

Beim Start eines Java-Programms wird die die Referenzvariable für ein Objekt aufgerufen und diese liegt auf dem Stack. Diese Referenzvariable verweist auf das Objekt.
Das eigentliche Objekt aber liegt, wie schon gesagt auf dem Heap.



Auf dem Heap wird ein Objekt solange bereitgestellt, bis es nicht mehr gebraucht oder bewusst zerstört wird.

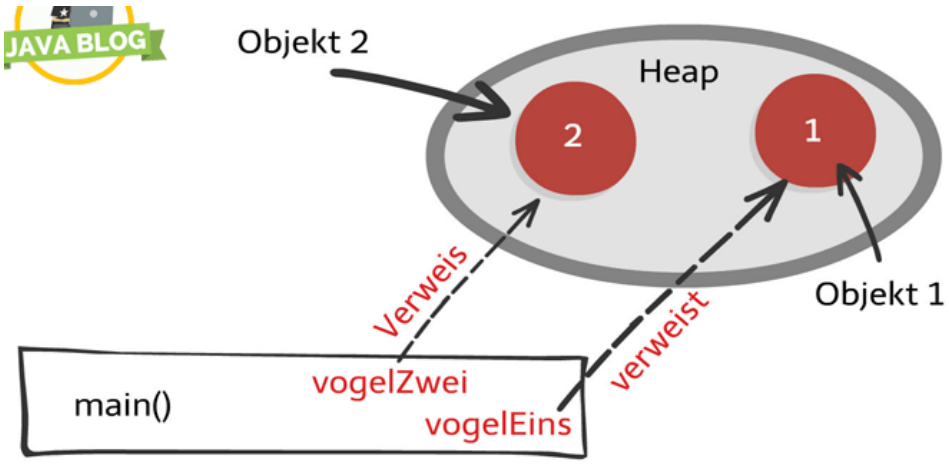
Wenn du Java Objekte aus dem Heap entfernen willst, schenke ihnen einfach keine Beachtung mehr. Hört sich blöd an, ist aber so.

Im Vogelspiel wird es so sein:

- Erster Vogel taucht auf und wird abgeschossen.
- Dann kommt der nächste Vogel ins Spiel.

Das heißt, ich lege einen zweites „Vogelobjekt“ und eine zweite Referenzvariable an.

Die Referenzvariable „vogelEins“ verweist auf das erste Objekt.
Die Referenzvariable „vogelZwei“ verweist auf das zweite Objekt.



Wenn der erste Vogel abgeschossen wurde, brauche ich diesen nicht mehr und ich kann den Verweis auf das erste Objekt entfernen und Objekt 1 steht nun ohne Referenzvariable da.

Und was passiert nun mit dem ersten Objekt?

Für Java Objekte ohne Verweis ist der Garbage Collector zuständig.
Der Garbage Collector ist die Müllabfuhr von Java, es sucht im ganzen Programm nach Objekten ohne Verweis und löscht diese aus dem Speicher.

Wie aktivierst du nun den Garbage Collector?

Gar nicht. Der Garbage Collector erkennt das automatisch:

Zusammenfassung:
- Java unterscheidet zwei wesentliche Speichersysteme.
- **Objekte und Instanzvariablen** werden im **Heap** verwaltet.
- **Methoden und deren lokalen Variablen** befinden sich im **Stack-Speicher**.
- Wenn du in **ein neues Objekt** anlegst, befindet sich die **Referenzvariable** (Verweis) im **Stack** und das **Objekt** im **Heap**.
- Um den Heap zu bereinigen, musst du nur den Verweis entfernen, denn **Objekte ohne Verweis** werden durch den **Garbage Collector** automatisch aus dem Heap entfernt.